Preface: Special Section on Formal Methods for Industrial Critical Systems (Selected Papers from FMICS'11)

This section contains extended versions of selected papers from the 16$^{\text{th}}$ International Workshop on Formal Methods for Industrial Critical Systems (FMICS'11).

The FMICS workshop series provides a forum for researchers who are interested in the development and application of formal methods in industry. In particular, FMICS brings together scientists and engineers that are active in the area of formal methods and interested in exchanging their experiences in the industrial usage of these methods. The FMICS workshop series also strives to promote research and development for the improvement of formal methods and tools for industrial applications. FMICS'11 was the 16$^{\text{th}}$ workshop in this series, and was held in Trento (Italy) on August 29-30, 2011.

This year, we received 39 submissions at FMICS'11. After the review process, the international Program Committee decided to select 16 papers for presentation during the workshop and inclusion in the FMICS'11 proceedings. From these 16 papers, the authors of seven papers were invited to submit an extended version to this special section. These extended papers went through a rigorous peer review process. The revised versions of five papers were finally accepted and are included in this special section. We believe that the papers presented here provide key insights on different aspects of industrial systems verification, such as runtime verification, testing, abstraction techniques, (bounded/symbolic) model checking, data flow analysis, or performance evaluation.

The first article in this special section, *"Runtime Verification of Microcontroller Binary Code"*, by Thomas Reinbacher *et al.*, presents a framework for runtime verification of microcontroller binary code, which aims at bridging the gap between formal verification and testing by providing techniques and tools that connect executions of a software to its specification. Their approach works in a non-intrusive fashion: The framework neither requires code instrumentation nor does it affect the execution of the analyzed program. This is achieved using a dedicated hardware unit that runs on a field programmable gate array in parallel to the analyzed microcontroller program. Different instances of this framework are discussed, with varying degrees of expressiveness of the supported specification languages and complexity in the hardware design. These instances range from invariant checkers for a restricted class of linear template constraints to a programmable processor that supports past-time linear temporal logic with timing constraints.

The second article, *"Formal Analysis of a Hardware Dynamic Task Dispatcher with CADP"*, by Etienne Lantreibecq and Wendelin Serwe, reports on the use of the CADP toolbox for the formal modeling and analysis of the Dynamic Task Dispatcher, a complex hardware block of an industrial hardware architecture developed by STMicroelectronics. The formal LNT model developed by an industry engineer was appropriate to discuss implementation details

with the architect and enabled model checking temporal properties expressed in MCL, which revealed a possible problem. The existence of the problem was investigated in the architect's C++ model using co-simulation of the C++ and the formal LNT models.

The third article, *"Observations on Formal Safety Analysis in Practice"*, by Michaela Huhn and Stefan Milius, reports on the application of formal verification in the safety analysis of two level crossing controllers that were industrially designed using the SCADE Suite. Although the theoretical grounds for formalizing safety analysis have been developed in recent years, numerous and intense complexity problems even with these medium-sized industrial case studies still have to be faced. The complexity problems constricted formal verification and even remained after employing different heuristics based on abstraction and introducing environmental models. In addition, it was observed that the modeling style has a significant impact on the complexity of the verification tasks. Finally, all relevant fault combinations were successfully classified as either critical or uncritical by identifying a crucial, design-specific liveness property.

The fourth article, *"A Case Study on the Lightweight Verification of a Multi-Threaded Task Server"*, by Néstor Cataño *et al.*, presents a case study on the verification of the design of a commercial multi-threaded task server (MTTS), developed by the Novabase company, used for massively parallelising computational tasks. In a first stage, the Plural tool was employed to perform lightweight verification of Java programs using a Data Flow Analysis (DFA) framework, to specify and verify the MTTS. In a second stage, the Pulse tool that enhances the analysis performed by Plural, was developed and used on the MTTS specifications. Pulse translates Plural specifications into an abstract state-machine model that captures the semantics of all the possible concurrent programs implementing the given specifications, and uses the evmdd-smc symbolic model checker to verify the machine model. The experimental results on the MTTS specification show that the exhaustive model checking approach scales reasonably well and is efficient at finding errors in specifications that were not previously detected with the DFA capabilities of Plural.

The fifth article, *"Experiences with Formal Engineering: Model-Based Specification, Implementation and Testing of a SoftwareBus at Neopost"*, by Marten Sijtema *et al.*, reports on the actual industrial use of formal methods during the development of a software bus. This work was achieved in two phases: a first phase in which the software bus was developed at Neopost Inc., and a second, post-case study phase, where model checking of the bus protocol was performed and the quality and performance of the model-based testing process measured. The approach of using formal methods in both the design step and the integration testing step of the V-model was a success: with a relatively limited effort, five subtle bugs were found.

Many people have contributed to this special section, without whose effort this special section would not have been possible. Besides the authors of the papers, we would like to thank the members of the Program Committee of the workshop: María Alpuente (Technical University of Valencia, Spain), Jiri Barnat (Masaryk University, Czech Republic), Josh Berdine (Microsoft Research,

Cambridge, UK), Jan Olaf Blech (fortiss GmbH, Germany), Rance Cleaveland (Reactive Systems, USA), Cindy Eisner (IBM, Israel), Wan Fokkink (Vrije Universiteit Amsterdam, Netherlands), Stefania Gnesi (ISTI-CNR, Italy), Holger Hermanns (Universitt des Saarlandes, Germany), Daniel Kästner (AbsInt GmbH, Germany), Stefan Kowalewski (RWTH-Aachen University, Germany), Daniel Kroening (University of Oxford, UK), Frédéric Lang (INRIA Rhône-Alpes, France), Kim G. Larsen (Aalborg University, Denmark), Diego Latella (ISTI-CNR, Italy), Timo Latvala (Space Systems Finland), Corina Pasareanu (NASA Ames, USA), Charles Pecheur (University of Louvain, Belgium), Ernesto Pimentel (University of Málaga, Spain), Jaco van de Pol (Universiteit Twente, The Netherlands), Marco Roveri (FBK-IRST, Italy), John Rushby (SRI International, USA), Thomas Santen (Microsoft European Innovation Center, Germany), Marjan Sirjani (Reykjavik University, Iceland), Helmuth Veith (TU Wien, Austria). Our thanks go also to the anonymous referees who kindly agreed to help us with the selection and reviewing of the papers in this special section and carried out an excellent job during this lengthy and laborious process.

Gwen Salaün
Grenoble INP, Inria, France

Bernhard Schätz
fortiss GmbH, Germany

Science of Computer Programming Guest Editors